Synergia Update October 2009

James Amundson and Alexandru Macridin

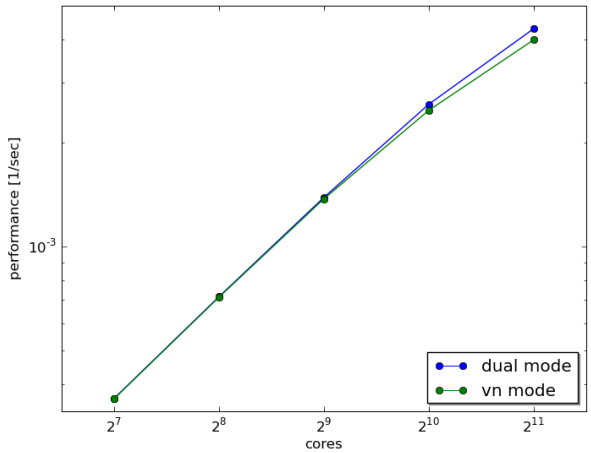
Fermilab

ComPASS Collaboration Meeting, October 6, 2009

Outline

Recent Synergia developments

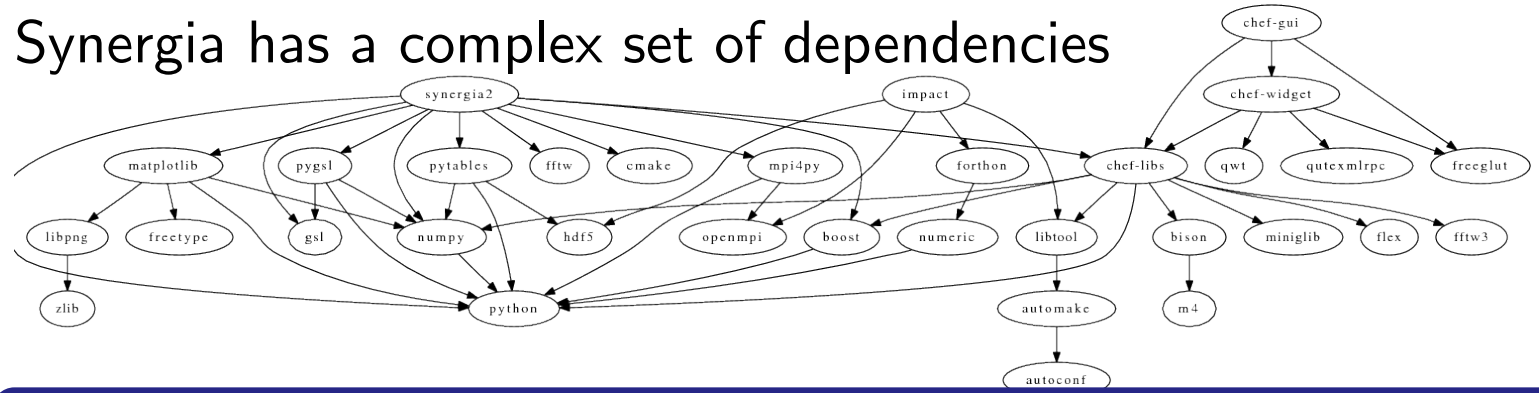
- Physics: resistive wall impedance
  - See Alex Macridin's slides...
- Infrastructure: porting/portability
  - Synergia2 ported to BG/P (Argonne's Intrepid)
    - Scaling to 2048 processors
    - Build still an expert-level process (but not for long...)
  - Contractor updated
    - Installed packages now auto-detectable
    - Advanced options allow (nearly) arbitrary end-user command modifications



Synergia scaling study performed at ALCF with 200 million particles with space charge calculated on a $64 \times 64 \times 1024$ *grid*

Contractor update

Synergia has a complex set of dependencies



Synergia build instructions (non-cross compiling platforms)

```
./contract.py
```

- Installed packages are auto-detected
- Build can be stopped and restarted
- Zero buy-in: packages themselves not modified

Synergia build instructions for BlueGene/P (soon!)

```
./contract.py --configure-import bgp.txt
./contract.py
```

Applications

BeamBeam3d

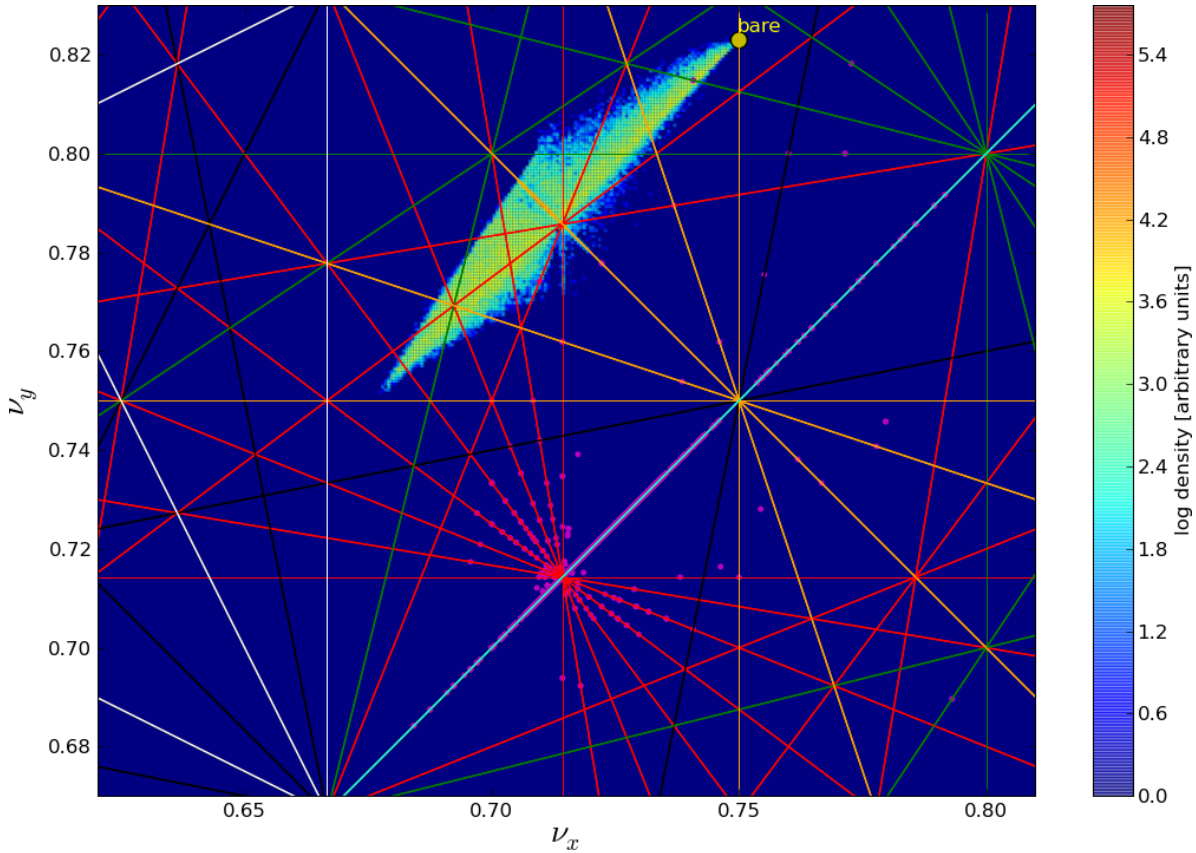
- Paper *Fully 3D multiple beam dynamics processes simulation for the Fermilab Tevatron* submitted to Physical Review ST-AB
- No new BeamBeam3d simulations planned

Synergia

- Mu2e
  - Simulations of resonant extraction from Debuncher including space charge
  - Program well underway; some results shown at ICAP09
  - Space charge is probably a limiting factor for this experiment
- Project X
  - Simulations of space charge effects in the Main Injector
  - Program just started
  - Resistive wall and multi-bunch effects after space charge
  - Will be comparing against IMPACT simulations

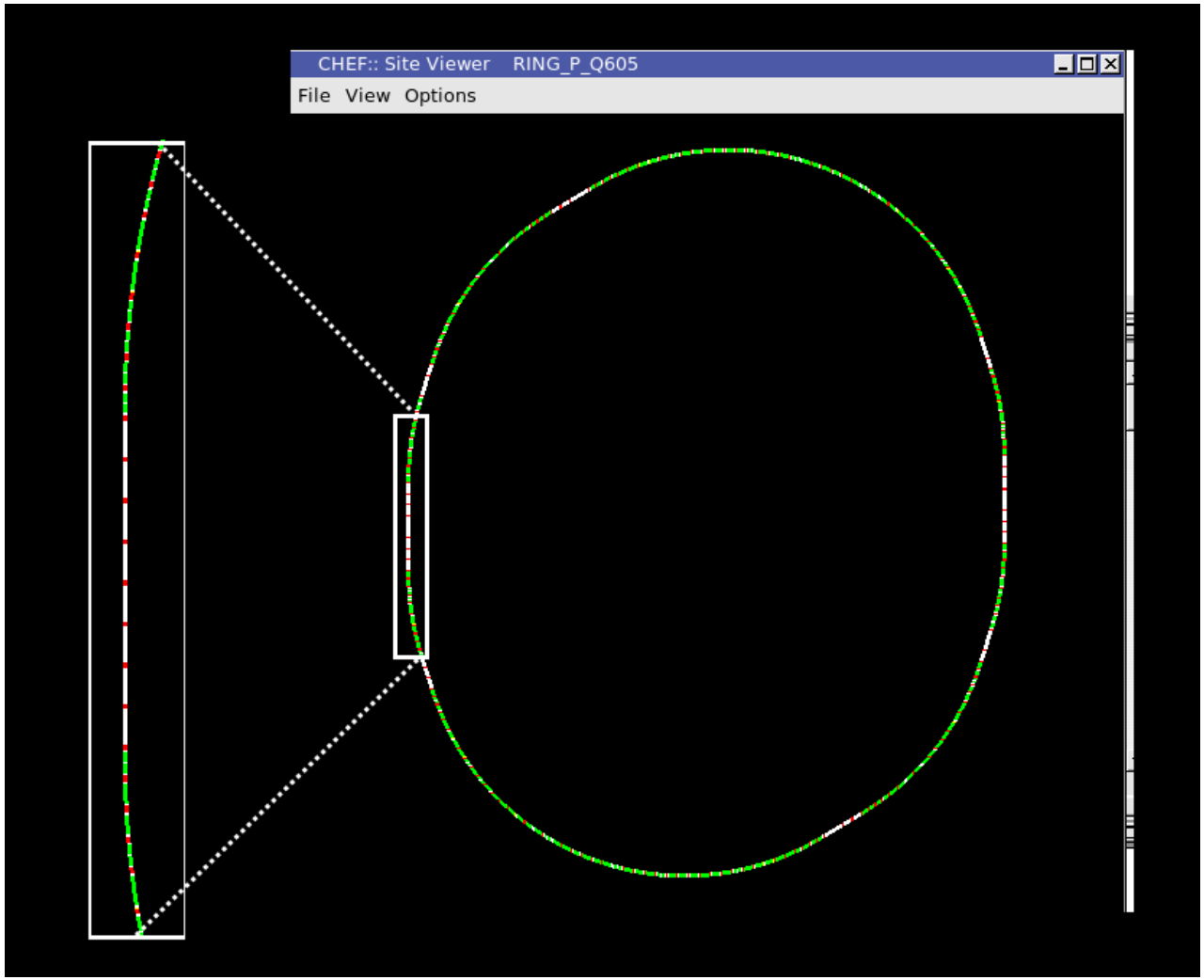
Mu2e

Highlight from Mu2e simulations



- Tune footprint (colors) and losses (points)
- Resonance lines included up to seventh order

Fermilab Main Injector (Project X)



Development goals

Public release of Synergia

- Project server `http://compacc.fnal.gov/projects`
  - Redmine, which is currently spreading throughout Fermilab
- Interface cleanup
- Documentation
  - Using Sphinx (Python) and Doxygen (C++) via Breathe

Scalability to 10^4+ processors

- Massively parallel multi-bunches
  - Main Injector has 588 buckets
  - INCITE proposal
- Algorithmic advances for massively parallel space charge
  - See Alex Macridin's slides
  - Also part of INCITE proposal

Development details

- Interface cleanup
  - Prefer pure C++ and pure Python modules
  - Unit tests
    - Boost::Test (C++) and Nose (Python)
  - Inline documentation
    - ..as an automatically included part of hand-written documentation
  - Bunch- and solver-specific communicators
    - Necessary for massively parallel bunches
  - Standardize on Boost::MultiArray
- CHEF improvements
  - In-progress code review of CHEF
    - Expert review team includes member of the C++ Standards Committee
    - Results expected soon
    - Implementation of recommendations to follow

Discussion question

Have you benchmarked GNU compilers *vs.* IBM compilers for your code on BG/P?

My tests showed IBM losing big in some cases, winning small in others. In all cases, compiling with IBM takes *several times* as long as compiling with GNU. I was not compelled to try to get all dependent packages compiling with IBM compilers.